(19)日本国特許庁(JP)　(12)公表特許公報（A）　(11)特許出願公表番号

特表平10－512699

(43)公表日　平成10年(1998)12月2日

(51)Int.Cl.[6]　識別記号　FI

G06F　17/30　G06F　15/40　310　F

15/419　320

審査請求　有　予備審査請求　有　(全38頁)

(21)出願番号　特願平9-522215

(86)(22)出願日　平成8年(1996)12月10日

(85)翻訳文提出日　平成10年(1998)3月24日

(86)国際出願番号　PCT/US96/19831

(87)国際公開番号　WO97/22069

(87)国際公開日　平成9年(1997)6月19日

(31)優先権主張番号　08/571,748

(32)優先日　1995年12月13日

(33)優先権主張国　米国（US）

(71)出願人　ディジタル　イクイプメント　コーポレイション
アメリカ合衆国　マサチューセッツ州　01754　メイナード　パウダーミル　ロード　111

(72)発明者　モーニア　ルイス　エム
アメリカ合衆国　カリフォルニア州　94061　レッドウッド　シティー　メリーランド　ストリート　2019

(74)代理人　弁理士　中村　稔　(外6名)

最終頁に続く

(54)【発明の名称】コンピュータのネットワークからワールドワイドウェッブ上のページを捜し出したり、ドキュメントを捜し出したりするためのシステム及び方法

(57)【要約】

迅速に、ネットワークによって接続されているコンピュータからワールドワイドウェッブ上のウェッブページをフェッチして解析するためのウェッブクローラーシステム及び方法であり、ランダムアクセスメモリ（RAM）に記憶されたハッシュテーブル及びシーケンシャルウェッブインフォメーションディスクファイルを含む。システムに既知である全てのウェッブページについて、ウェッブクローラーシステムは、ハッシュテーブルにより小さなエントリーを記憶するのに加えて、シーケンシャルディスクファイルにエントリーを記憶する。ハッシュテーブルエントリーは、識別値、対応するウェッブページが上手くフェッチされた時だけ真がセットされるフェッチフラグ、対応するエントリーがシーケンシャルディスクファイルのどこに記憶されているかを示すファイル位置インジケータを含む。シーケンシャルディスクファイルのエントリーの各々は、対応するウェッブページのURL及びそのウェッブページに関するフェッチステータス情報を含む。ウェッブインフォメーションディスクファイルへの全てのアクセスは入力バッファを経由してシ

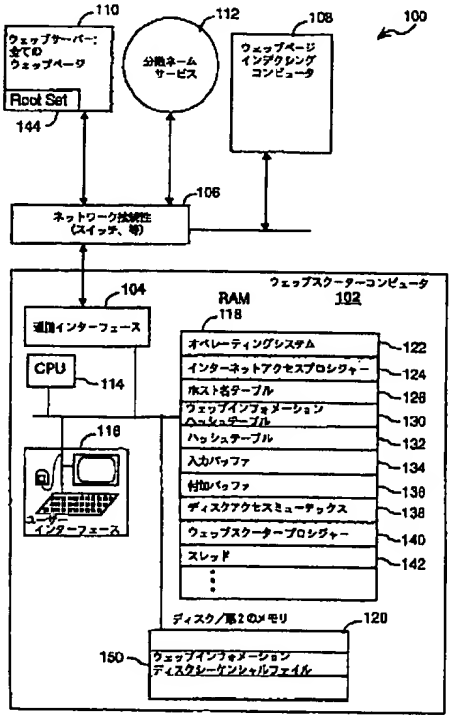

FIG. 1

【特許請求の範囲】

１．各ウェッブページは固有のＵＲＬ（ユニバーサルリソースロケータ）を有し、少なくともいくつかの前記ウェッブページは他のウェッブページへのＵＲＬリンクを含んでいるような、遠隔地に配置されたアクセス可能なコンピュータに記憶されているウェッブページを含むデータセットを捜し出すためのシステムであり、

対応するＵＲＬに従って、前記の遠隔地に配置されたコンピュータから特定のウェッブページをフェッチするための通信インターフェースと、

エントリーの各々が対応するウェッブページのＵＲＬ及びフェッチステータス情報を示すような一セットのエントリーを有するウェッブインフォメーションファイルと、

ＲＡＭ（ランダムアクセスメモリ）に記憶されていて、エントリーの各々が、対応するウェッブページの識別値及びフェッチステータス情報を示すようなエントリー一セットを有するウェッブインフォメーションテーブルと、

ウェッブインフォメーションファイルのエントリーが前記フェッチステータス情報に基づく事前に規定された選定規準を満たすようなウェッブページをフェッチするための命令と、受け取られた各々のウェッブページ中の各々のＵＲＬリンクについて、ウェッブインフォメーションテーブル中に対応するエントリーが既に存在するかどうかを決定し、ウェッブインフォメーションテーブルに対応するエントリーを有していないＵＲＬリンクの各々について、ウェッブインフォメーションテーブルに新しいエントリーを加え、ウェッブインフォメーションファイルに対応する新しいエントリーを加えるための命令とを含む、前記システムによって実行される、ウェッブページをフェッチして解析するためのウェッブスクータープロシジャーを実行する手段と

を備えるシステム。

２．重複する時間期間中に、各々がウェッブスクータープロシジャーを実行するような多数のスレッドを含み、スレッドのいくつかがウェッブページをフェッチしている間に、ウェッブページの他のスレッドは、フェッチされたウェッブ

ページを解析しているような手段を含む請求項１に記載のシステム。

３．ミューテックスを含み、スレッドの各々によって実行される前記ウェッブスクータープロシジャーは、ウェッブインフォメーションテーブル及びウェッブインフォメーションファイルにアクセスする前にミューテックスを要求して待つための命令を含む請求項２に記載のシステム。

４．入力バッファ及び付加バッファと、

シーケンシャルに並べられたエントリーのブロックをウェッブインフォメーションファイルから入力バッファへ記憶するためのファイルマネージャーと、

入力バッファ中のウェッブインフォメーションファイルのエントリーをスキャンして解析し、前記の事前に規定された選定規準を満たす前記ウェッブインフォメーションファイルのエントリーを捜し出す前記ウェッブスクータープロシジャーと、

前記ウェッブインフォメーションファイルに加えられるべき全てのエントリーを前記付加バッファに記憶する前記ウェッブスクータープロシジャーと、付加バッファ中の多数のエントリーをウェッブインフォメーションファイルに移すための前記ファイルマネージャーと

を含む請求項３に記載のシステム。

５．第２のメモリー中のエントリーの各々は第１のメモリー中の対応するエントリーのアドレスを含む請求項１に記載のシステム。

６．各ウェッブページは固有のＵＲＬ（ユニバーサルリソースロケーター）を有し、少なくともいくつかの前記ウェッブページは他のウェッブページへのＵＲＬリンクを含んでいるような、遠隔地に配置されているがアクセス可能なコンピュータに記憶されているウェッブページを含むデータセットを捜し出す方法であり、

各エントリーが、対応するウェッブページのＵＲＬ及びフェッチステータス情報を示すような一セットのエントリーを有するウェッブインフォメーションファイルを記憶するステップと、

各エントリーが、対応するウェッブページの識別値及びフェッチステータス情報を示すような一セットのエントリーを有するウェッブインフォメーション

テーブルをＲＡＭ（ランダムアクセスメモリ）に記憶するステップと、

（Ａ）ウェッブインフォメーションファイル中のエントリーをシーケンシャルにスキャンして、前記エントリーのどれが事前に規定された選定規準を満たすかを決定し、（Ｂ）ウェッブインフォメーションファイルのエントリーが前記の事前に規定された選定規準を満たすようなウェッブページをフェッチし、（Ｃ）受け取られたウェッブページの別のウェッブページへのＵＲＬリンクの各々について、対応するエントリーが既にウェッブインフォメーションテーブル中に存在するかどうかを決定し、（Ｄ）ウェッブインフォメーションテーブル中に対応するエントリーを有していないＵＲＬリンクの各々について、ウェッブインフォメーションテーブルに新しいエントリーを付加し、ウェッブインフォメーションファイルに対応する新しいエントリーを付加することを含む、ウェッブページをフェッチして解析するためのウェッブスクータープロシジャーを実行するステップと

を備える方法。

７．重複する時間期間中に多数のスレッドにおいて前記ウェッブスクータープロシジャーを実行し、スレッドのいくつかがウェッブページをフェッチしている間に、ウェッブページの他のスレッドはフェッチされたウェッブページを解析するようにすることを含む請求項６に記載の方法。

８．ミューテックスを定義し、

前記スレッドの各々において前記ウェッブスクータープロシジャーを実行している間に、ウェッブインフォメーションテーブル及びウェッブインフォメーションファイルにアクセスする前に、ミューテックスを要求して待つこと

を含む請求項７に記載の方法。

９．前記ＲＡＭに、「入力バッファ」及び「付加バッファ」を定義し、

シーケンシャルに並べられたエントリーのブロックをウェッブインフォメーションファイルから入力バッファへ記憶し、

シーケンシャルにウェッブインフォメーションファイルのエントリーをスキャンする前記のステップは、入力バッファのウェッブインフォメーションファイルのエントリーをスキャンして、前記ウェッブインフォメーションファイル

のエントリーのどれが前記の事前に規定された選定規準を満たすかを決定することを含むステップを備え、

前記ファイルに加えられるべき全てのエントリーを前記付加バッファに記憶し、

付加バッファの多数のエントリーをウェッブインフォメーションファイルに移す

ステップを備えている請求項８に記載の方法。

10．ウェッブインフォメーションテーブルのエントリーの各々はウェッブインフォメーションファイルの対応するエントリーのアドレスを含み、

ウェッブインフォメーションテーブルの対応するエントリーのアドレスを読み出して、それから前記アドレスにある前記ウェッブインフォメーションファイルの前記の１エントリーを読み出すことによって、前記ウェッブインフォメーションファイルの前記エントリーの１つにアクセスすること

を含む請求項６に記載の方法。

11．各データセットはアドレスによって固有に識別され、少なくともいくつかのデータセットは、コンピュータに記憶された他のデータセットの接続アドレスを１つ以上含むような、ネットワークによって接続されたコンピュータに記憶されているデータセットを捜し出すための装置であり、

識別されたデータセットの要求をコンピュータに送り、前記の要求に応答してデータセットを受け取るための、ネットワークに接続された通信インターフェースと、

各々が対応するデータセットのアドレス及び対応するデータセットのステータス情報を含んでいるようなエントリーの第１の一セットを記憶している第１のメモリと、

各々が対応するデータセットのアドレスの符号化及び対応するデータセットのステータス情報の符号化を含んでいるようなエントリーの第２の一セットを記憶している第２のメモリと、

第１と第２のメモリ及び通信インターフェースに接続され、シーケンシャルに第１の一セットのエントリーを読み出し、事前に規定されたステータスに基

づく選定規準を満たすような対応するエントリーを第１の一セット中に有する識別されたデータセットの要求をつくり、識別されたデータセットを受け取るのに応答して、前記の第１及び第２の一セットに、第２の一セットに対応するエントリーが存在しない受け取られたデータセットの少なくともアドレスの集合の各々に対応する新しいエントリーをつくるスレッド手段と

を備える装置。

12．第２の一セットのエントリーの各々は第１の一セットの対応するエントリーのアドレスを含み、エントリーの前記の第２の一セットはエントリーの第１の一セットにインデックスを付けるためのものであるような請求項１１に記載の装置。

13．スレッド手段のいくつかが前記の要求をつくり、識別されたデータセットを受け取っている間に、他のスレッド手段は前記第１と第２のメモリに新しいエントリーをつくっているような多数の前記スレッドを含む請求項１１に記載の装置。

14．ミューテックスを含み、前記スレッド手段の各々は、第１のメモリ及び第２のメモリにアクセスする前にミューテックスを要求して待つロジックを含むような請求項１３に記載の装置。

15．前記第２のメモリに配置された入力バッファ及び付加バッファと、

第１のメモリのシーケンシャルに並べられたエントリーのグループを入力バッファに記憶するマネージャーと、

入力バッファのエントリーをスキャンして解析し、前記の事前に規定されたステータスに基づく選定規準を満たす前記エントリーを捜し出す手段を含む前記スレッド手段の各々と、

前記第１のメモリに加えられるべき全てのエントリーを前記付加バッファに記憶する前記スレッド手段の各々と、

付加バッファの多数のエントリーを第１のメモリに移す手段も有する前記マネージャーと

を含む請求項１４に記載の装置。

16．各データセットはアドレスによって固有に識別され、少なくともいくつかの

前記データセットは、コンピュータに記憶された他のデータセットの接続アドレスを１つ以上含むような、ネットワークによって接続されたコンピュータに記憶されたデータセットを捜し出す方法であり、

（Ａ）各々が対応するデータセットのアドレス及び対応するデータセットのステータス情報を含んでいるような、エントリーの第１の一セットを第１のメモリに記憶するステップと、

（Ｂ）各々が対応するデータセットのアドレスの符号化及び対応するデータセットのステータス情報の符号化を含んでいるような、エントリーの第２の一セットを第２のメモリに記憶するステップと、

（Ｃ）シーケンシャルに第１の一セットのエントリーを読み出すステップと、

（Ｄ）事前に規定されたステータスに基づく選定規準を満たす、第１の一セットの対応するエントリーを有する識別されたデータセットの要求を、ネットワークを経由してコンピュータに伝送するステップと、

（Ｅ）識別されたデータセットを受け取るのに応答して、前記第１及び第２の一セットに、第２の一セットに対応するエントリーが存在しない少なくとも受け取られたデータセットのアドレスの集合の各々に対応する新しいエントリーをつくるステップ

を備える方法。

17．前記ステップＢは、第２の一セットのエントリーの各々に、第１の一セットの対応するエントリーのアドレスを記憶し、エントリーの前記の第２の一セットはエントリーの第１の一セットにインデックスを付けるためのものであるような請求項１６に記載の方法。

18．重複する時間期間中に多数のスレッドにおいてステップＣ、Ｄ、Ｅを実行し、スレッドのいくつかがデータセットをフェッチしている間に、データセットの他のスレッドは、フェッチされたデータセットを解析しているようにすることを含む請求項１６に記載の方法。

19．ミューテックスを定義し、

前記スレッドの各々は、第１及び第２のメモリのエントリーの第１及び第２の一セットにアクセスする前にミューテックスを要求して待つこと

を含むような請求項１８に記載の方法。

20．前記第２のメモリに入力バッファ及び付加バッファを定義し、

シーケンシャルに並べられたエントリーのブロックをエントリーの第１の一セットから入力バッファに記憶し、

前記のシーケンシャルに読み出すステップは、入力バッファのエントリーをシーケンシャルに読み出し、前記入力バッファのエントリーのどれが前記の事前に規定されたステータスに基づく選定規準を満たすかを決定するステップを備え、

前記第１のメモリに加えられるべき全てのエントリーを前記付加バッファに記憶し、

付加バッファの多数のエントリーを第１のメモリに移すこと

を含む請求項１９に記載の方法。

【発明の詳細な説明】

コンピュータのネットワークからワールドワイドウェッブ上のページを捜し出したり、ドキュメントを捜し出したりするためのシステム及び方法

発明の分野

本発明は、一般的に、コンピュータのネットワークからワールドワイドウェッブ（WWW）上のページと呼ばれるドキュメントにアクセスしたり、ドキュメントを捜し出したりするためのシステム及び方法に関し、特に、迅速にワールドワイドウェッブ上のページを捜し出して解析するためのシステム及び方法に関する。

発明の背景

ここではウェッブページと呼ぶウェッブドキュメントは、インターネットに接続された多数のサーバーコンピュータ（ここでは以後「サーバー」と呼ぶ。）上に記憶される。ウェッブ上の各々のページは別個のＵＲＬ（ユニバーサルリソースロケーターuniversal resource locator）を有する。ウェッブサーバー上に記憶された多数のドキュメントはＨＴＭＬ（ハイパーテキストマークアップランゲージーhypertext markup language）と呼ばれる標準のドキュメント記述言語で書かれている。ＨＴＭＬを使用して、ウェッブドキュメントのデザイナーは、ドキュメント中でハイパーテキストリンクもしくはアノテーションをドキュメント中の特定の語又は句と関連付けて、ウェッブページの視覚的な外観及び内容を記述する。ハイパーテキストリンクは、その語又は句に関する情報を提供する他のウェッブドキュメントもしくは同一のドキュメント中の他の部分のＵＲＬを識別する。

インターネットに接続されたウェッブクライアント上で動くウェッブブラウザー（ＨＴＭＬドキュメントを表示したり、ウェッブサーバーと通信したりするように設計されたコンピュータプログラム）を使用して、ユーザーはWWW上に記

憶されたドキュメントにアクセスする。一般的に、ユーザーが、ウェッブブラウザーで表示されるドキュメント内でハイパーテキストリンク（一般的に、強調された語もしくは句としてウェッブブラウザーで表示される。）を選ぶことによっ

て、ユーザはドキュメントにアクセスする。そして、ウェッブブラウザーは、要求されたドキュメントのＵＲＬによって識別されるウェッブサーバーへ、要求されたドキュメントのＨＴＴＰ（ハイパーテキスト転送プロトコルーhypaertext transfer protocol）要求を出す。その要求に応答して、やはりＨＴＴＰを使用して、指名されたウェッブサーバーは要求されたドキュメントをウェッブブラウザーに返す。

１９９５年の末以後、ワールドワイドウェッブ（以後、「ウェッブ」と呼ぶ。）として知られているインターネット部分のページ数は、先の１年間に数倍にもなり、少なくとも３０００万ページに達するようになった。本発明は、ウェッブが増大し続ける時に、ウェッブ上のページの経路を維持し続けるためのシステムを実施することを意図している。

ウェッブ上のページを捜し出すためのシステムは、「ウェッブクローラー（Web crawler）」や「ウェッブスパイダー（Web spider）」や「ウェッブロボット（Web robot）」として様々に知られている。本発明は「ウェッブスクーター（Web scooter）」としてつくられてきた。何故ならば、それは既知のウェッブクローラーのどれよりも非常に速いからである。本文では、「ウェッブクローラー」、「ウェッブスパイダー」、「ウェッブスクーター」、「ウェッブクローラーコンピュータシステム」、「ウェッブスクーターコンピュータシステム」という語を相互に互換性を有する語として使用する。

一般的に、従来技術のウェッブクローラーは以下のように動作する。既知のウェッブページのルートセットから開始して、全ての既知のウェッブページに対する別個のエントリーについて、ディスクファイルがつくられる。更なるウェッブページがフェッチされ、他のページへのそれらのリンクが解析される時、まだウェッブクローラーに知られていないウェッブページを参照するために、ディスクファイルに更なるエントリーがつくられる。エントリーの各々は、他のステータス情報と共に、対応するウェッブページが処理されているかどうかを示す。ウェッブクローラーはウェッブページを次のように処理する。（Ａ）処理されているページ中の他のウェッブページへの全てのリンクを識別し、関連する情報を記憶

して、まだ処理されていない識別された全てのウェッブページを、処理されるべきウェッブページのリスト、もしくは他の同等のデータ構造に加える。（Ｂ）ウェッブページをインデクサ、もしくは他のドキュメント処理システムに送る。

一般的に、既に処理されたウェッブページに関する情報はディスクファイルに記憶される。何故ならば、ディスクファイルの情報量は、ランダムアクセスメモリ（ＲＡＭ）に記憶するには大きすぎるからである。例えば、もし平均１００バイトの情報がウェッブページのエントリーの各々に対して記憶されるならば、３０００万ウェッブページを表すデータファイルは約３ギガバイトになり、これはＲＡＭに実際に記憶するには大きすぎる。

次に、１ウェッブページを処理する時に発生するディスクＩ／Ｏについて考える。この説明のために、典型的な１ウェッブページは２０個の他のウェッブページへのリファレンスを有し、ディスク記憶装置は１秒あたり５０シークよりも多くの処理はできないと仮定する。ウェッブクローラーは、処理されているページ中の２０個のページリファレンスの各々を評価して、ウェッブクローラーがそれらのページについて既に知っているかどうかを決定しなければならない。これを実施するために、ウェッブクローラーはウェッブインフォメーションディスクファイルから２０個のレコードの検索を試みなければならない。もし、特定ページのリファレンスのレコードが既に存在するならば、そのリファレンスは捨てられる。何故ならば、更なる処理は不要だからである。しかしながら、もし特定ページのレコードが見つからないならば、そのページのアドレスの可能なエイリアスの各々に対してレコードを捜し出す試みがなされなければならない。それによって、標準の１ウェッブページを解析するのに必要なディスクレコードの平均シーク数は、１ページあたり約５０ディスクシークにまで増大する。

特定ページのリファレンスのディスクファイルのレコードがまだ存在していないならば、参照されたページの新しいレコードがつくられてディスクファイルに加えられる。そして、そのページリファレンスが、処理されるべきページの待ち行列に加えられるか、もしくはそのページがまだフェッチされて処理されていないことを示すのに、そのディスクファイルエントリ自体が使用される。

このように、単に１ウェッブページを処理するのに、（存在するレコードを読み出し、新しいレコードを書き込むために）おおよそ２０ディスクシークが必要となる。結果として、１秒あたり５０ディスクシークの制限を与えられているので、１秒あたり約１ウェッブページしか処理されない。

加えて、ネットワークアクセス待ち時間の問題がある。ウェッブサーバー及びウェッブサーバーとウェッブクローラーコンピュータ上の両方で使用される特定のハードウェアとソフトウェアの位置によって、ウェッブページを検索する時間は大きく変わるけれども、平均的に、ウェッブページを検索するのには約３秒かかる。このように、ネットワーク待ち時間もやはり、従来技術のウェッブクローラーによって処理されるウェッブページ数を１秒あたり約０．３３ウェッブページに制限する恐れがある。ディスク「シーク」制限、ネットワーク待ち時間、及び他の遅延要因のために、代表的な従来技術のウェッブクローラーは１日あたり約３０，０００ウェッブページよりも多くのページを処理することができない。

ウェッブページがウェッブに追加される速度、及びウェッブページが削除されたり、改訂されたりする速度の理由から、１日あたり３０，０００ウェッブページの処理では、ウェッブ上の全てのウェッブページの真に最新のディレクトリーもしくはインデックスを維持するには不十分である。理想的には、ウェッブクローラーは１日あたり少なくとも２５０万ウェッブページを訪れる（すなわち、フェッチ及び解析する）ことができなければならない。

従って、非常に高速の性能を持つウェッブクローラが必要となる。本発明の目的は、１日あたり数百万のウェッブページを処理することができるように改良されたウェッブクローラーを提供することである。本発明の関連する目的は、主に、ウェッブクローラーのＣＰＵの処理速度によってのみ、ウェッブクローラーの動作速度が制限されるようにするために、前記のディスク「シーク」制限及びネットワーク待ち時間制限を解決するような改良されたウェッブクローラーを提供することである。更に、本発明の別の関連する目的は、平均して、１秒あたり少なくとも３０ウェッブページ、より好ましくは１秒あたり少なくとも１００ウェッブページをフェッチ及び解析することができるウェッブクローラーシステムを提

供することである。

発明の概要

本発明の本質は、請求項１に記述されているようなウェッブページを捜し出すためのシステム及び請求項６に記述されているようなウェッブページを捜し出すための方法に存在する。

以後に、迅速に、ワールドワイドウェッブ上のウェッブページのディレクトリーを捜し出して作成するためのシステム及び方法を説明する。ウェッブクローラーシステムは、ランダムアクセスメモリ（ＲＡＭ）に記憶されたハッシュテーブル及び代表的なディスク記憶装置である第２のメモリに記憶されたシーケンシャルファイル（ここでは、「シーケンシャルディスクファイル」もしくは「ウェッブインフォメーションディスクファイル」と呼ぶ。）を含む。システムにとって既知である全てのウェッブページに対して、ウェッブクローラーシステムは、ハッシュテーブルにより小さなエントリーを記憶するのに加えて、シーケンシャルディスクファイルにエントリーを記憶する。ハッシュテーブルのエントリーは、識別値、対応するウェッブページが上手くフェッチされた時だけ真にセットされるフェッチフラグ、対応するエントリーがシーケンシャルディスクファイルのどこに記憶されているかを示すファイル位置インジケーターを含む。シーケンシャルディスクファイルエントリーの各々は、対応するウェッブページのＵＲＬとそのウェッブページに関するフェッチステータス情報を含む。

ウェッブインフォメーションディスクファイルへの全てのアクセスは、単一のＩ／Ｏオペレーションとして、シーケンシャルディスクファイルから多数のエントリを入力バッファに移すといったように、入力バッファを経由して、シーケンシャルに行われる。従って、シーケンシャルディスクファイルは入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられる全ての新しいエントリーは付加バッファに記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容はシーケンシャルディスクファイルの最後に加えられる。このようにして、ウェッブインフォメーションディスクファイルへのランダムア

クセスは排除され、ディスクアクセス制限によって引き起こされる待ち時間は最小化される。

ウェッブページを捜し出して、そのページを処理するためのプロシジャーは、シーケンシャルにシーケンシャルファイルの全てのエントリーを見直し、設定された選定規準を満たす次のエントリーを選ぶことを含む。処理する次のファイルエントリーを選ぶ時、ハッシュテーブルは、最新のエントリー候補の全ての既知のエイリアスと照合されて、エイリアスでそのウェッブページが既にフェッチされているかどうかを決定する。もしエイリアス下でそのウェッブページは既にフェッチされているならば、そのシーケンシャルファイルのエントリーのエラータイプフィールドは「非ー選定エイリアス」として記録され、その候補エントリーは選ばれない。

一度、次のウェッブページのリファレンスエントリーが選ばれると、ウェッブクローラーシステムは対応するウェッブページへのフェッチを試みる。もし、フェッチが不成功ならば、そのウェッブページのシーケンシャルファイルエントリーのフェッチステータス情報は、ウェッブクローラーに返されたエラーリターンコードに従ってフェッチ失敗として記録される。もし、フェッチが成功ならば、そのウェッブページの（入力バッファ中の）シーケンシャルディスクファイルのエントリーの類似のフェッチフラグと同様に、そのウェッブページのハッシュテーブルエントリーのフェッチフラグがセットされる。加えて、フェッチされたウェッブページ中のＵＲＬリンクの各々が解析される。もしそのリンクによって参照されるＵＲＬもしくはそのＵＲＬの規定されたエイリアスのいずれかのエントリーが既にハッシュテーブルにあるならば、そのＵＲＬリンクの更なる処理は必要ではない。もしこのようなエントリーがハッシュテーブル中に見つからないならば、そのＵＲＬは、まだウェッブページのウェッブクローラのデータベースに含まれていない「新しい」ウェッブページを表しており、従って、新しいウェッブページのエントリーがシーケンシャルディスクファイルに加えられる（すなわち、そのエントリーは付加バッファのディスクファイルの一部に加えられる。）。その新しいディスクファイルのエントリーは処理されているリンクによって参照されるＵＲＬを含み、「未フェッチ」と登録される。加えて、対応する新しい

エ

ントリーがハッシュテーブルに加えられ、そしてそのエントリーのフェッチフラグはクリアされて、対応するウェッブページはまだフェッチされていないことを示すようにする。フェッチされたページ中の全てのＵＲＬリンクの処理に加えて、更なる処理のために、ウェッブクローラーはフェッチされたページをインデクサーに送る。

図面の簡単な説明

添付図を参照して例示された以下の好ましい実施態様の説明によって、本発明のより詳細な理解が得られるであろう。

・図１は本発明の好ましい実施態様に従うウェッブクローラーシステムの好ましい実施態様のブロック図である。

・図２は本発明の好ましい実施態様で使用されるハッシュテーブルの仕組みのブロック図である。

・図３は本発明の好ましい実施態様で使用されるシーケンシャルなウェッブインフォメーションディスクファイル及び関連するデータ構造のブロック図である。

・図４は本発明の好ましい実施態様で使用されるウェッブクローラープロシジャーのフローチャートである。

好ましい実施態様の説明

図１について、ウェッブスクーターコンピュータシステム１０２を含む分散コンピュータシステム１００が示されている。通信インターフェース１０４及び一セットのインターネットや他のネットワークへの接続１０６によって、ウェッブスクーターはインターネットやウェッブページインデクシングコンピュータ（Web page indexing computer）１０８に接続される。ある実施態様では、ウェッブページインデクシングコンピューター１０８は、ローカルもしくはワイドエリアネットワーク接続を使用せずに、専用通信チャネルを通って、ウェッブスク

ーター１０２に直接接続される。ウェッブスクーター１０２が接続されるインタ

ーネット部分は、（Ａ）ウェッブページを記憶するウェッブサーバー１１０及び（Ｂ）ここでは総体的に参照番号１１２で参照されている、分散ネームサービス（ＤＮＳ）として知られているサービスに協力するサーバーである。本文では、ＤＮＳ１１２は、全てのインターネットのホスト名に対して規定された全てのエイリアスーセットを全ての要求者に提供し、そしてインターネットのホスト名及びそれらのエイリアスは、全てのＵＲＬの先頭部分を形成すると仮定する。

好ましい実施態様では、ウェッブスクーター１０２はデジタルイクイップメント社製のアルファワークステーションコンピューターであるが、実際は、あらゆるタイプのコンピューターをウェッブスクーターコンピュータとして使用することができる。好ましい実施態様では、ウェッブスクーター１０２はＣＰＵ１１４、前記の通信インターフェース１０４、ユーザーインターフェース１１６、ランダムアクセスメモリ（ＲＡＭ）１１８、ディスクメモリ（ｄｉｓｋ）１２０を含む。好ましい実施態様では、通信インターフェース１０４は非常に高性能の通信インターフェースであり、１秒あたり少なくとも３０ウェッブページの平均フェッチスループットで、１０００以上の重複する通信要求を取り扱うことができる。

好ましい実施態様では、ウェッブスクーターのＲＡＭは１ギガバイトのランダムアクセスメモリを有し、以下のものを記憶する。

・マルチタスクオペレーティングシステム１２２。

・ＤＮＳ１１２からエイリアス情報をフェッチするためであるのに加えて、ウェッブページをフェッチするためでもあるインターネット通信マネージャープログラム１２４。

・ホスト名に対して規定されたエイリアスを表す情報を記憶するホスト名テーブル１２６。

・ウェッブインフォメーションハッシュテーブル１３０。

・ハッシュテーブルマネージャープロシジャー１３２。

・入力バッファ１３４及び付加バッファ１３６。

・ハッシュテーブル１３０、入力バッファ１３４、付加バッファ１３６へのアクセスを制御するためのミューテックス（ｍｕｔｅｘ）１３８。

・ウェッブスクータープロシジャー１４０。

・Ｔ１スレッドの実行を決定するためのスレッドデータストラクチャー１４２。

ここで、Ｔ１の値はウェッブスクーターコンピューターシステム１０２のオペレーターが選ぶことができる整数である。（例えば、好ましい実施態様では、Ｔ１は１０００の値にセットされる。）

より詳細を以下に説明するように、ディスク記憶装置１２０は、入力バッファ１３４及び付加バッファ１３６を経由して、シーケンシャルにアクセスされるウェッブインフォメーションディスクファイル１５０を記憶する。

ホスト名テーブル１２６は、特に、ＤＮＳ１１２にとって既知である各ホスト名の全てのエイリアスを表す情報を記憶する。エイリアスは、ウェッブスクータープロシジャー１４０によって特定のウェッブページのＵＲＬのホスト名部分の代わりにされる効果的なＵＲＬの先頭部分の一セットであり、特定のウェッブページのエイリアスのＵＲＬの一セットを成す。

次に、上記データ構造及びプロシジャーの使用及び動作を図１－図４及び表１－表２を参照して説明する。表１－表２は共にウェッブスクータープロシジャーの疑似コード表現を含む。ここで使用される疑似コードは、この説明の目的のためだけにつくられているが、その疑似コードは一般的なコンピュータ言語の規約を使用しており、当業者である全てのコンピュータプログラマーが容易に理解可能であるように設計されている。

ウェッブインフォメーションハッシュテーブル

図２に関して、ウェッブインフォメーションハッシュテーブル１３０は、フェッチされて解析されたウェッブページのＵＲＬリンクによって参照される各ウェッブページに加えて、ウェッブスクーターシステムによってフェッチされて解析された各ウェッブページに対する別個のエントリー１６０を含む。このようなエントリーの各々は、以下を含む。

・対応するウェッブページに固有な識別値１６２。

・対応するウェッブページがウェッブスクーターによってフェッチされて解析

されたかどうかを示す１ビットの「フェッチフラグ」１６４。

・ウェッブインフォメーションディスクファイル１５０の対応するエントリーの位置を示すファイル位置値１６６。

好ましい実施態様では、識別値の各々は６３ビットの長さであり、ファイル位置値は各々３２ビットの長さである。結果として、好ましい実施態様において、ハッシュテーブルエントリーの各々は丁度１２バイトを占める。ハッシュテーブルエントリーのサイズそのものは重要でないが、ハッシュテーブルエントリー１６０の各々は対応するディスクファイルエントリーよりもかなり小さい（例えば、平均して少なくとも７５％小さい。）ということは重要である。

ハッシュテーブルマネージャー１３２は、その「インターフェース」１７０を経由して、ウェッブスクータープロシジャー１４０から以下の２種類のプロシジャーコールを受け取る。

・第１の要求はハッシュテーブルマネージャー１３２に特定のＵＲＬのエントリーが存在するかどうかを問い合わせ、そして、もし存在するならば、その記録のフェッチフラグが、対応するウェッブページが先にフェッチされて解析されたことを示すかどうかを問い合わせる。

・第２の要求は、特定のＵＲＬ及び特定のディスクファイル位置の新しいエントリーをハッシュテーブル１３０に記憶するようにハッシュテーブルマネージャーに要求する。

ハッシュテーブルマネージャー１３２は識別ハッシュ関数１７２を使用して、そこに現れる全てのＵＲＬの６３ビットの識別値を計算する。識別関数１７２は、確実に全ての固有のＵＲＬが同様に固有の識別値に変換されるように設計されている。識別関数は全ての固有のウェッブページのＵＲＬの圧縮されたコードをつくる。通常の当業者であれば、適切な識別関数の設計を理解している。約2^{25}から2^{26}のウェッブページがあると、識別値は2^{63}の別個の値を持つことが可能であるということを注記する。

ハッシュテーブルが既に固有のＵＲＬのエントリーを有するかどうかを、ウェッブスクータープロシジャー１４０がハッシュテーブルマネージャー１３２に問い合わせる時、ハッシュテーブルマネージャーは、（Ａ）前記の識別ハッシュ関

数１７２を使用して、固有のＵＲＬの識別値をつくる。（Ｂ）ハッシュテーブル１３０のどこにその識別値を有するエントリーを記憶するかを決定するハッシュテーブル位置関数１７４にその値を送る。（Ｃ）実際に、このようなエントリーがハッシュテーブルに記憶されているかどうかを決定する。（Ｄ）もしマッチするエントリーが見つからないならば、失敗値（例えば、－１）を返す。（Ｅ）もしハッシュテーブルにそのエントリーが見つかったならば、成功値（例えば、０）及びそのエントリーのフェッチフラグ値とディスク位置値を返す。

好ましい実施態様では、識別値の所定数の低位ビットに基づいて、ハッシュテーブル位置関数１７４はハッシュテーブルエントリーの位置を決定し、同一の低位ビットを持つ全ての識別値のエントリーのブロックのチェーンに続く。ハッシュテーブル１３０中の、与えられた値の低位ビットのエントリー１６０は、１ブロックあたりＢ１エントリーのブロックに配置される。ここで、Ｂ１は調整可能なパラメーターである。好ましい実施態様で使用される上記の方法は、ハッシュテーブル１３０に高密度な方法でデータを記憶するという利点がある。当業者は理解しているように、多くの他のハッシュテーブル位置関数を使用することができる。

ウェッブスクータープロシジャー１４０が、ハッシュテーブルマネージャー１３２に特定のＵＲＬ及び特定のディスクファイル位置の新しいハッシュテーブルのエントリーを記憶することを要求する時、ハッシュテーブルマネージャーは、（Ａ）前記の識別ハッシュ関数１７２を使用して、特定のＵＲＬの識別値をつくる。（Ｂ）ハッシュテーブル１３０のどこに識別値を有するエントリーを記憶しなければならないかを決定するハッシュテーブル位置関数１７４にその値を送る。（Ｃ）ハッシュテーブルの所定の位置に、対応するウェッブページがまだフェッチされていないことを示すフェッチフラグ値、識別値、特定のディスクファイル位置と共に新しいエントリー１６０を記憶する。

ウェッブインフォメーションディスクファイル及びバッファ

図３及び表２に関して、入力バッファ１３４及び付加バッファ１３６は、どちらのバッファもＲＡＭに配置されており、これらのバッファの使用によって、デ

ィスクアクセスの動作は最小化される。入力バッファ及び付加バッファの管理は、ディスクファイルマネージャーとしても知られているバックグラウンドのシーケンシャルディスクファイル及びバッファハンドラープロシジャーによって実施される。

好ましい実施態様では、入力バッファ及び付加バッファはサイズが各々５０から１００メガバイトである。入力バッファ１３４は、ウェッブインフォメーションディスクファイル１５０の、シーケンシャルに並べられた連続部分を記憶するのに使用される。ウェッブスクータープロシジャーは、入力バッファ１３４、付加バッファ１３６、ディスクファイル１５０の使用の調整を要求される多数の他のブックキーピングポインター（bookkeeping pointer）に加えて、入力バッファの処理されるべき次のエントリーへのポインター１７６、及びウェッブインフォメーションディスクファイル１５０の入力バッファ１３４に転送されるべき次のエントリー１８０へのポインター１７８を維持する。

単一のＩ／Ｏ動作として、多数のエントリーがシーケンシャルディスクファイルから入力バッファへ移されるといったように、ウェッブインフォメーションディスクファイル１５０への全てのアクセスは、入力バッファ１３４を経由して、シーケンシャルに行われる。従って、シーケンシャルディスクファイル１５０は入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられる全ての新しいエントリーは付加バッファ１３６に記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容がシーケンシャルファイルの最後に加えられる。このようにして、ウェッブインフォメーションディスクファイルへのランダムアクセスは減少され、ディスクアクセス制限によって引き起こされる待ち時間は最小化される。

ウェッブスクーターによって入力バッファ１３４の全てのエントリーがスキャンされる度に、入力バッファのエントリーへの全ての更新がウェッブインフォメーションディスクファイル１５０に再度記憶され、付加バッファ１３６の全てのエントリーがディスクファイル１５０の最後に加えられる。加えて、付加バッファ１３６はクリアされ、ディスクファイルのエントリーの次の一セットが、（ポ

インター１７８によって示される）入力バッファ１３４にコピーされるべきエントリーの最後の一セットの直後から、入力バッファ１３４にコピーされる。ウェッブスクロータープロシジャーによってディスクファイルの最後のエントリーがスキャンされると、スキャンはディスクファイル１５０の先頭に戻る。

付加バッファ１３６が新しいエントリーで一杯になった時はいつでも、その中身はディスクファイル１５０の最後に加えられ、そして、付加バッファはクリアされて新しいエントリーを受け取る。

ウェッブインフォメーションディスクファイル１５０のエントリー１８０の各々は以下を記憶する。

・エントリーによって参照されるウェッブページのＵＲＬを記憶する可変長のＵＲＬフィールド１８２。

・ウェッブスクローターによって、対応するウェッブページがフェッチされて解析されたかどうかを示すフェッチフラグ１８４。

・参照されたウェッブページがフェッチされ、解析され、そしてインデックスを付けられた日付及び時間を示すタイムスタンプ１８６。

・ウェッブページのサイズを示すサイズ値１８８。

・もし何か、エントリーが重複している（すなわち、エイリアスのＵＲＬの）エントリーで、無視されるべきであるといったような場合に、又は、参照されるウェッブページにフェッチする最後の試みが行われた時に発生したエラータイプを示すエラータイプ値１９０。

・ここでは取り上げない他のフェッチステータスパラメーター１９２。

ＵＲＬフィールド１８２は可変長であるので、ウェッブインフォメーションディスクファイル１５０のレコード１８０もまた可変長である。

ウェッブスクロータープロシジャー

図１－図４及び表１の疑似コードに関して、好ましい実施態様におけるウェッブスクロータープロシジャー１４０は以下の通り動作する。ウェッブスクロータープロシジャーが実行を開始する時、そのプロシジャーはシステムのデータ構造を初期化する（２００）。

・既に存在しているウェッブインフォメーションディスクファイル１５０をスキャンし、シーケンシャルファイルの全てのエントリーに対するエントリーについてハッシュテーブル１３０を初期化する。

・シーケンシャルディスクのエントリーの第１のバッチをディスクファイル１５０から入力バッファ１３４にコピーする。

・空の付加バッファ１３６を新しいシーケンシャルファイルのエントリー用に定義する。

・入力バッファ１３４、付加バッファ１３６、ハッシュテーブル１３０へのアクセスを制御するためのミューテックス１３８を定義する。

それから、ウェッブスクーターイニシャライザーはＴ１個のスレッドを開始し（例えば、好ましい実施態様では１０００個のスレッドが開始される。）、スレッドの各々は同一のスクータープロシジャーを実施する。

ウェッブスクーターイニシャライザープロシジャーの実施の前に既に存在しているウェッブインフォメーションディスクファイル１５０のエントリー一セットは既知のウェッブページの「ルートセット」１４４と呼ばれる。「アクセス可能」なウェッブページの一セットは、ルートセット中のＵＲＬリンクによって参照される全てのウェッブページ及び他のアクセス可能なウェッブページ中のＵＲＬリンクによって参照される全てのウェッブページから成る。このように、いくつかのウェッブページはウェッブスクーター１０２にとってアクセス不可であるようにすることが可能である。何故ならば、ルートセットと「アクセス不可な」ウェッブページとの間にはＵＲＬリンクがないからである。

様々なチャネルによって、このようなウェッブページに関する情報が使用可能になると、更なるエントリーの「マニュアル」挿入もしくは更なるエントリーを含むための他の仕組みによって、ウェッブインフォメーションディスクファイル１５０を拡張することができ（それによって、ルートセット１４４を拡張する。）、以前にアクセス不可であったウェッブページをアクセス可能にする。

以下は、全ての同時に実行されるスレッドによって実行されるウェッブスクータープロシジャーの説明である。プロシジャーの第１のステップはミューテック

ス（２０２）を要求して待つことである。ミューテックスの所有権が要求され、２つのスレッドが同一のディスクファイルのエントリーを処理しないように、そして２つのスレッドが同時にハッシュテーブル、入力バッファ、不可バッファもしくはディスクファイルへの情報の書き込みを試みないようにする。ハッシュテーブル１３０、入力バッファ１３４、付加バッファ１３６、ディスクファイル１５０はここでは総合的に「保護されたデータ構造」と呼ばれる。なぜならば、ミューテックスの使用によって、それらは総合的に保護されているからである。一度スレッドがミューテックスを所有すると、そのスレッドが規定された選定規準（２０４）を満たすエントリーを捜し出し、そのエントリーを選ぶまで、そのスレッドは入力バッファ中のディスクファイルのエントリーを（ポインター１７６によって示される）まだスキャンされていない次のエントリーからスキャンする（２０４）。

例えば、デフォルトの選定規準は次の通りである。「エントリーによって、一度もフェッチされていない、もしくは最後にフェッチされて解析されたのがＨ１時間よりも以前であると示されているようなウェッブページを参照する全てのエントリー。ここで、Ｈ１はオペレーターが選定可能な値である。ただし、エントリーは重複しているエントリーであることをエラータイプフィールドが示している（すなわち、以下に説明されるように「非－選定エイリアス」である。）エントリーを除く。」もしＨ１が１６８にセットされるならば、最後にフェッチされて解析されたのが１週間よりも前であるようなウェッブページを参照する全てのエントリーが選定規準を満たす。ウェッブページの大きさが考慮されるような選定規準の別の例は次の通りである。「一度もフェッチされていないウェッブページ、もしくは最後にフェッチされて解析されたのがＨ１時間よりも前であり、大きさがＳ１よりも大きいようなウェッブページ、もしくは最後にフェッチされて解析されたのがＨ２時間よりも前であり、大きさがＳ１以下であるようなウェッブページを表しているエントリー。ただし、エントリーが「非－選定エイリアス」であることをエラータイプフィールドが示しているエントリーを除く。ここでＳ１、Ｈ１、Ｈ２はオペレーターが選定可能な値である。」

処理すべき次のエントリーを選定する時、ハッシュテーブルを検索して現在の

エントリー候補の全ての既知のエイリアスを見つけ、エイリアスでそのウェッブページが既にフェッチされたかどうかを決定する。特に、もしエントリーが、規定された選定規準を満たすならば、ホスト名テーブル１２６の情報を使用して、そのエントリーのＵＲＬの全ての既知のエイリアスがつくられ、それからハッシュテーブル１３０が検索され、参照されたウェッブページがそのエイリアスのＵＲＬ下でフェッチされたことを示すフェッチフラグを有するエイリアスのＵＲＬのいずれかのエントリーを、そのハッシュテーブルが記憶しているかどうかを調べる。もし入力バッファ中の現在のエントリー候補によって参照されるウェッブページが、エイリアスのＵＲＬ下で既にフェッチされていると判断されるならば、その入力バッファのエントリーのエラータイプフィールド１９０は変更され、このエントリーは「非ー選定エイリアス」であると示すようにする。このようにして、今回及び以後、エントリーが更なる処理のために選定されるのを防ぐ。

一度、ウェッブページのリファレンスエントリーが選定されると、ミューテックスは解放され、他のスレッドが保護されたデータ構造にアクセス可能となる（２０６）。それから、ウェッブスクータープロシジャーは対応するウェッブページをフェッチするのを試みる（２０８）。フェッチが成功したか、もしくは失敗した後、再度、そのプロシジャーはミューテックスを要求して待ち（２１０）、再度、そのプロシジャーが保護されたデータ構造を使用できるようにする。

もしフェッチが不成功ならば（２１２－Ｎ）、そのウェッブページのシーケンシャルファイルのエントリー中のフェッチステータス情報を、ウェッブクローラーへ返されたエラーリターンコードに従ってフェッチ失敗として記録する（２１４）。もしフェッチが成功ならば（２１２－Ｙ）、（入力バッファの）シーケンシャルディスクファイルのエントリー１８０中のそのウェッブページのフェッチフラグ１８４のように、ハッシュテーブルのエントリー１６０中のそのウェッブページのフェッチフラグ１６４がセットされる。加えて、フェッチされたウェッブページのＵＲＬリンクの各々が解析される（２１６）。

フェッチされたウェッブページが解析された、もしくはフェッチ失敗が入力バッファのエントリーに記録された後、ミューテックスは解放され、他のスレッド

が保護されたデータ構造にアクセスできるようにする（２１８）。

次に、フェッチされたウェッブページのＵＲＬリンクを解析するためのプロシジャーを図４Ｂを参照して説明する。ウェッブページは、インデクシングシステム１０８によってインデックスを付けるための適切な情報を保有していない画像ファイルのようなドキュメントへのＵＲＬリンクを保有することができるということをここで注記する。しばしば、これらの参照されるドキュメントは、それらを参照するウェッブページの構成要素として使用される。本文では、画像ファイルや他のインデックス付け不可ファイルのような構成要素のファイルへのＵＲＬリンクは、「他のウェッブページへのＵＲＬリンク」とはしない。インデックス付け不可ファイルへのこれらのＵＲＬリンクは、ウェッブスクータープロシジャーによって無視される。

一度、他のウェッブページに接続する全てのＵＲＬを処理してしまうと（２３０）、インデックスを付けるためのインデクサーにフェッチされたウェッブページを送り（２３２）、ウェッブスクーターによる、フェッチされたウェッブページの処理を完了する。そうでない場合には、ウェッブページへの次のＵＲＬリンクが選定される（２３４）。もし選定されたリンクに関連するＵＲＬのハッシュテーブルのエントリーが既に存在するならば、そのリンクの更なる処理を要求せず、もし解析されているウェッブページにどれか未処理のＵＲＬリンクが残っているならば、次のＵＲＬリンクを選定する（２３４）。

もし選定されたリンクに関連するＵＲＬのハッシュテーブルのエントリーがまだ存在しないならば、ホスト名テーブル１２６の情報を使用して、そのエントリーのＵＲＬの全ての既知のエイリアスをつくる。それから、ハッシュテーブル１３０を検索し、そのテーブルが、そのエイリアスのＵＲＬのいずれかのエントリーを記憶しているかどうかを調べる（２３８）。もしハッシュテーブル中にそのエイリアスのＵＲＬのいずれかのエントリーが存在するならば、そのリンクの更なる処理を要求せず、そしてもし解析されたウェッブページにどれか未処理のＵＲＬリンクが残っているならば、次のＵＲＬリンクを選定する（２３４）。

もしハッシュテーブル中に選定されたリンクのＵＲＬもしくはそのエイリアスのいづれかのエントリーが見つからないならば、そのＵＲＬは、まだウェッブペ

ージのウェッブクローラーのデータベースに含まれていない「新しい」ウェッブページを表し、従って新しいウェッブページのエントリーが、付加バッファ中のディスクファイルの一部に加えられる（２４０）。その新しいディスクファイルのエントリーは処理されたリンクによって参照されるＵＲＬを含み、「未フェッチ」と記録される。加えて、対応する新しいエントリーがハッシュテーブルに加えられ、そしてそのエントリーのフェッチフラグはクリアされて、対応するウェッブページはまだフェッチされていないことを示すようにする（２４０）。それから、もしウェッブページ中にどれか未処理のＵＲＬリンクが存在するならば、ウェッブページの処理はウェッブページの次の未処理のＵＲＬリンクについて継続する。

目的及び動作が本文の範囲外であるようなプロシジャーによって、ウェッブインフォメーションディスクファイル１５０へのインデックスとして、ウェッブインフォメーションハッシュテーブル１３０は使用される。何故ならば、ハッシュテーブル１３０は、既知のウェッブページの各々のディスクファイル位置の値を保有するからである。いいかえると、ウェッブインフォメーションハッシュテーブル中の対応するエントリーのディスクファイルアドレスを最初に読み出し、それからそのアドレスにあるウェッブインフォメーションディスクファイルのエントリーを読み出すことによって、ウェッブインフォメーションディスクファイル中のエントリーはアクセスされる。

他の実施態様

好ましい実施態様のハッシュテーブル構造１３０の代わりに、バランスドツリー（balanced tree）、スキップリスト（skip list）といったような、ウェッブインフォメーションハッシュテーブル１３０の同一の属性を有する全てのデータ構造を使用することが可能である。

解法として、本発明は３つの基本の仕組みを使用して、従来技術のウェッブクローラーの速度制限を克服している。

第１に、どのウェッブページリンクがまだウェッブクローラーに知られていない新しいウェッブページを表しているかを決定するのに十分な情報を含むウェッ

ブページディレクトリテーブルが、ＲＡＭ中に記憶され、ディスクファイルにアクセスする必要なく、受け取られたウェッブページを解析できるようにしている。

第２に、より完全なウェッブページディレクトリはシーケンシャルな順番にだけアクセスされ、ディスクアクセスがウェッブクローラーの実施速度に重要な影響を持たない程度まで、実施されるディスクアクセスの数を減少するような大きな入力及び付加バッファによって、それらのアクセスを実施する。

第３に、ウェッブスクータープロシジャーを実行するための多数の同時にアクティブなスレッドを使用し、そしてウェッブサーバーへの、同様の数の同時の通信チャネルを操作可能な通信インターフェースを備えることによって、本発明はネットワークアクセス待ち時間によって引き起こされる遅延を避ける。

特に、多数のスレッドがウェッブページフェッチ要求に対する応答を待っている間に、他のスレッドは受け取ったウェッブページを解析している。同一のウェッブスクータープロシジャーを実施する多数のスレッドを使用することによって、受け取られたウェッブページを処理できるようになるためにミューテックスを待っている受け取られたウェッブページに関するスレッドの待ち行列が、平均的に存在するようである。また、ウェッブページのフェッチは、時間的にずれて行われる傾向にある。結果として、ウェッブスクーターは殆どウェッブページを受け取るために待っていたり、他にする仕事がないという状態にはならない。マルチプロセッサーのワークステーションを使用し、そしてウェッブスクータープロシジャーを同時に実行するスレッドの数を更に増大することによって、ウェッブスクーターのスループットを更に増大することが可能である。

いくつかの特定の実施態様を参照して本発明を説明したが、この説明は本発明の例であり、本発明を限定するものとして解釈されてはならない。ここで提示され、請求された本発明の範囲から逸脱することなく、さまざまな改修が可能である。

表１

ウェッブスクーターブロシジャーの疑似コード表現

プロシジャー：ウェッブスクーター
｛
／＊初期化ステップ＊／
既に存在しているウェッブインフォメーションディスクファイル中をスキャンし、シーケンシャルファイル中の全てのエントリーに対するエントリーについてハッシュテーブルを初期化する。
シーケンシャルディスクのエントリーの第１のバッチをＲＡＭの入力バッファ中に読み出す。
新しいシーケンシャルファイルのエントリー用の空の付加バッファを定義する。
入力バッファ、付加バッファ、ハッシュテーブルへのアクセスを制御するためのミューテックスを定義する。
１０００スレッドを開始する。各々のスレッドは同一のスクーターブロシジャーを実行する。
｝

プロシジャー：スクーター
｛
ドゥ　フォーエバー（Do Forever）：
　　｛
　　ミューテックスを要求して待つ。
　　設定されたＵＲＬ選定規準に従って、処理するべき新しいＵＲＬが選定されるまで、（入力バッファの）シーケンシャルファイルを読み出す。処理する次のＵＲＬを選定する時、ハッシュテーブルを調べてＵＲＬの全ての既知のエイリアスを検索し、そのウェッブページがエイリアスで既にフェッチされているかどうかを決定する。そして、もしそのウェッブ

ページがエイリアスでフェッチされているならば、シーケンシャルファイルのエントリーのエラータイプフィールドに「非－選定エイリアス」として記録する。

／＊選定規準の例：ＵＲＬは一度もフェッチされていないか、もしくは最後にフェッチされたのはＨ１時間よりも前であり、かつ非－選定エイリアスではない。＊／

ミューテックスを解放する。

選定されたウェッブページをフェッチする。

ミューテックスを要求して待つ。

もしフェッチが成功ならば、

{

ハッシュテーブルのエントリー及び入力バッファのシーケンシャルファイルのエントリーにおいてページをフェッチ済として記録する。

／＊フェッチされたページを解析する。＊／

ページのＵＲＬリンクの各々に対して

{

もしＵＲＬもしくはいずれかの規定されたエイリアスが既にハッシュテーブルに存在するならば、

{何も行わない。}

他の場合には、

{

／＊そのＵＲＬは、まだデータベースに保有されていない「新しい」ウェッブページを表す。＊／

「未フェッチ」と記録されたエントリーと共に、対応するウェッブページの新しいエントリーを付加バッファに加える。

「未フェッチ」と記録されたエントリーと共に、エントリーをハッシュテーブルに加える。

}

```
        }
        処理するために、フェッチされたページをインデクサーに送る。
        }
    他の場合には、
        {
        現在処理されている入力バッファ中のエントリーに、受け取ったリターンコードに基づく適切な「フェッチ失敗」エラーインジケータを記録する。
        }
    ミューテックスを解放する。
    }    ／＊ドゥ　フォーエバーループの終了＊／
}
```

表２

バックグラウンドシーケンシャルファイルバッファハンドラーの
疑似コード表現

プロシジャー：バックグラウンドシーケンシャルファイルバッファハンドラー（ａ／ｋ／ａ　ディスクファイルマネージャー）
｛
「読み出しシーケンシャルファイル」命令が入力バッファをオーバーフローさせる時はいつでも、
　｛
　入力バッファをシーケンシャルディスクファイルに再コピーする。
　次のエントリー一セットを入力バッファに読み出す。
　付加バッファの内容をシーケンシャルディスクファイルの最後に付加する。
　付加バッファをクリアして、新しいエントリーに備える。
　｝
「シーケンシャルファイルへのエントリーの付加」が付加バッファのオーバーフローを引き起こす時はいつでも、
　｛
　付加バッファの内容をシーケンシャルディスクファイルの最後に付加する。
　付加バッファをクリアして、新しいエントリーに備える。
　処理が終了していない新しいエントリーを付加バッファの先頭に加える。
　｝
｝

【図１】

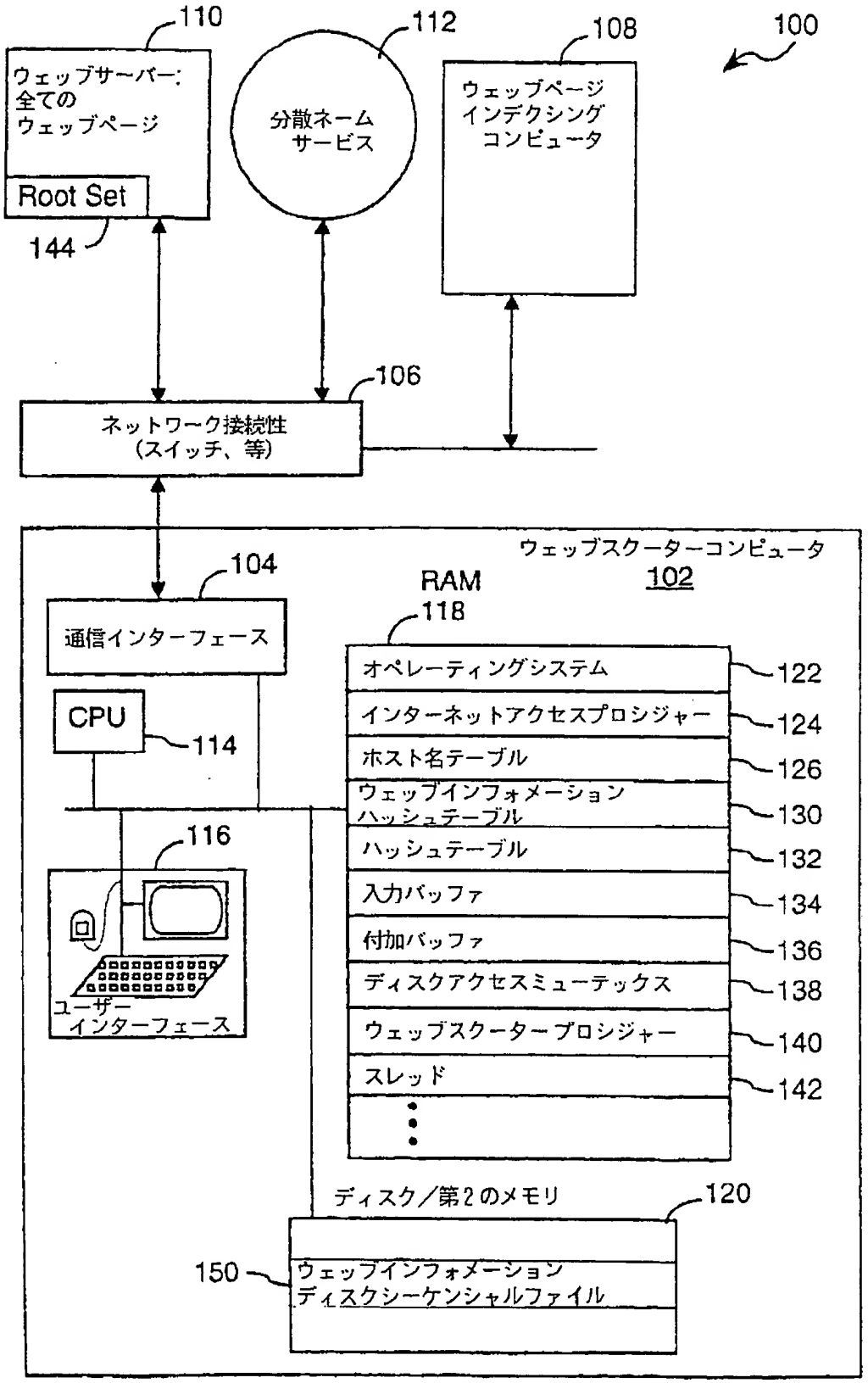
110
ウェッブサーバー:
全ての
ウェッブページ
Root Set
144
112
分散ネーム
サービス
108
ウェッブページ
インデクシング
コンピュータ
100
106
ネットワーク接続性
(スイッチ、等)
ウェッブスクーターコンピュータ
102
104
通信インターフェース
CPU
114
RAM
118
オペレーティングシステム
122
インターネットアクセスプロシジャー
124
ホスト名テーブル
126
ウェッブインフォメーション
ハッシュテーブル
130
ハッシュテーブル
132
入力バッファ
134
付加バッファ
136
ディスクアクセスミューテックス
138
ウェッブスクータープロシジャー
140
スレッド
142
116
ユーザー
インターフェース
ディスク/第2のメモリ
120
150
ウェッブインフォメーション
ディスクシーケンシャルファイル

FIG. 1

【図２】

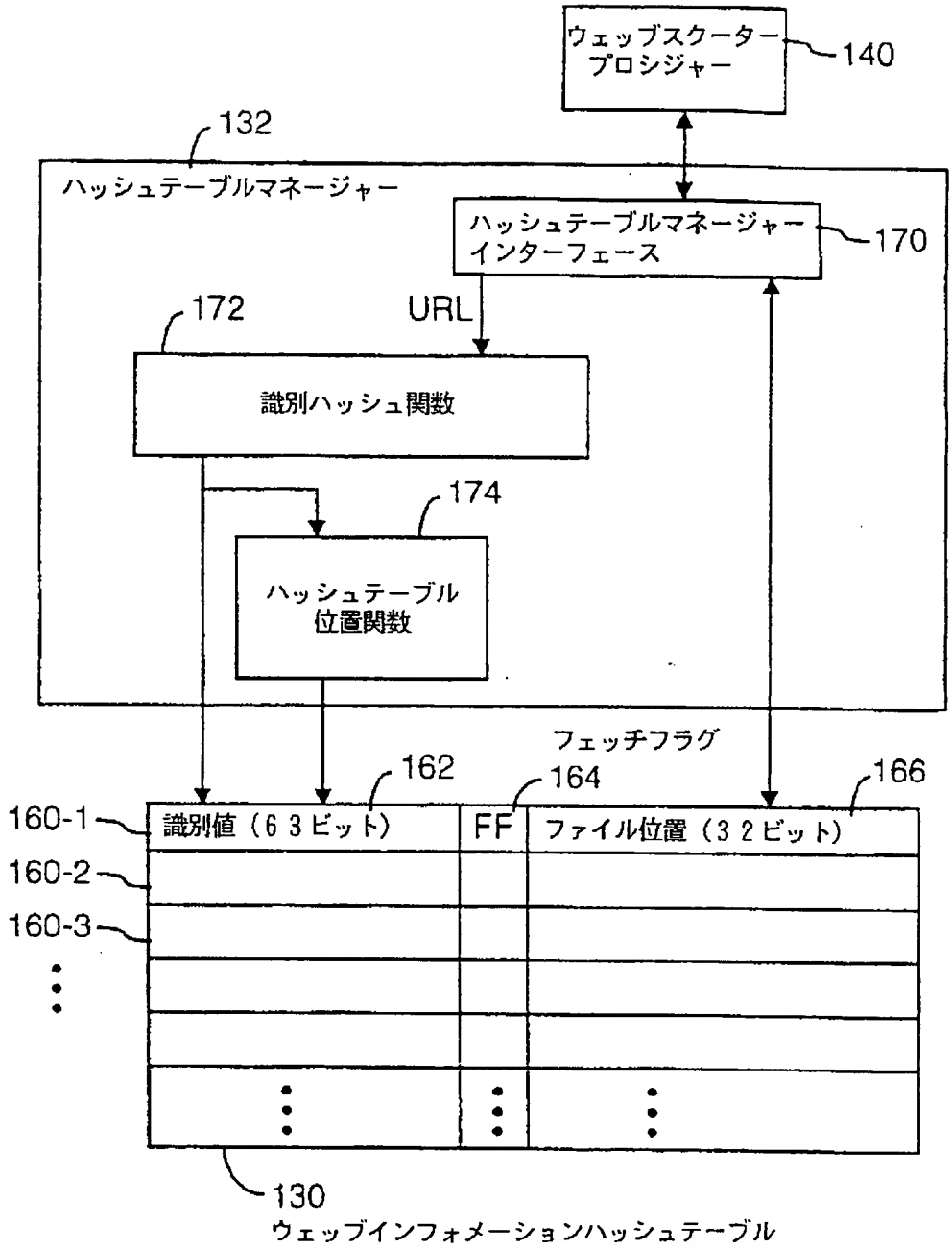
ウェッブスクーター
プロシジャー
140
132
ハッシュテーブルマネージャー
ハッシュテーブルマネージャー
インターフェース
170
172
URL
識別ハッシュ関数
174
ハッシュテーブル
位置関数
フェッチフラグ
162
164
166
160-1
160-2
160-3
識別値（６３ビット）
FF
ファイル位置（３２ビット）
130
ウェッブインフォメーションハッシュテーブル

FIG. 2

【図３】

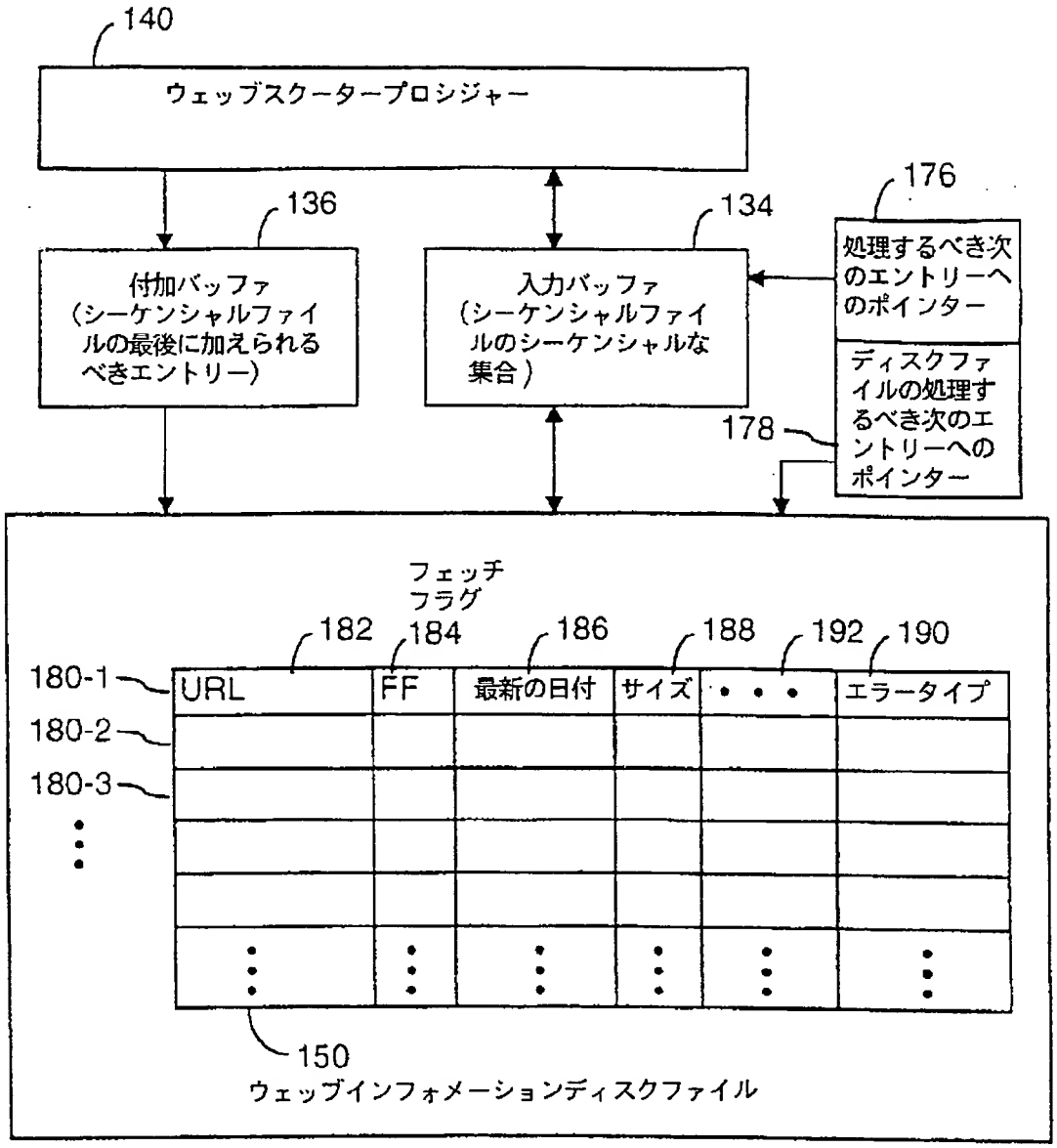
140
ウェッブスクータープロシジャー
136
付加バッファ
（シーケンシャルファイルの最後に加えられるべきエントリー）
134
入力バッファ
（シーケンシャルファイルのシーケンシャルな集合）
176
処理するべき次のエントリーへのポインター
ディスクファイルの処理するべき次のエントリーへのポインター
178
フェッチフラグ
182
184
186
188
192
190
180-1
180-2
180-3
URL
FF
最新の日付
サイズ
エラータイプ
150
ウェッブインフォメーションディスクファイル

FIG. 3

【図４】

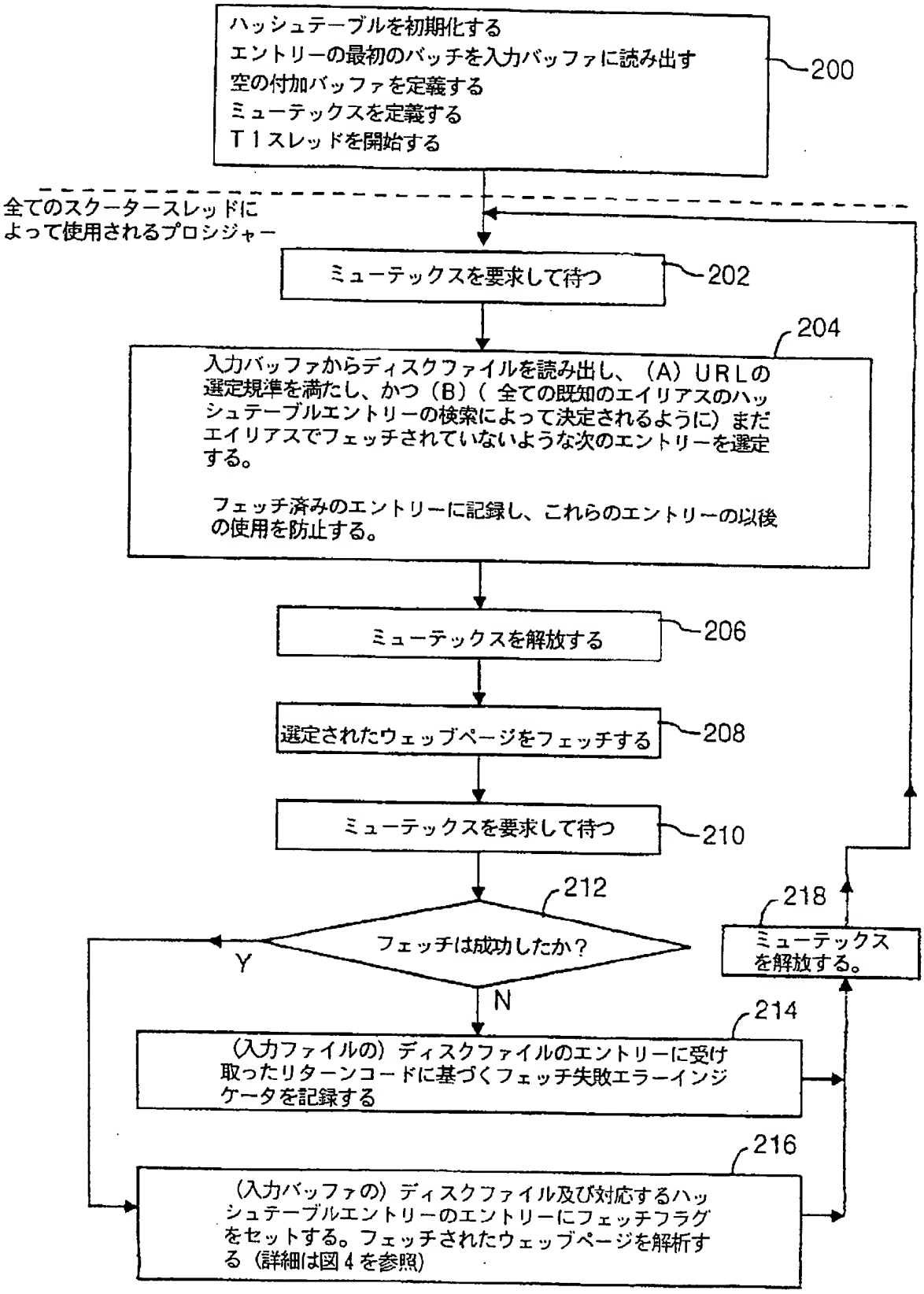
ハッシュテーブルを初期化する
エントリーの最初のバッチを入力バッファに読み出す
空の付加バッファを定義する
ミューテックスを定義する
Ｔ１スレッドを開始する
200
全てのスクータースレッドによって使用されるプロシジャー
ミューテックスを要求して待つ
202
204
入力バッファからディスクファイルを読み出し、（A）URLの選定規準を満たし、かつ（B）（全ての既知のエイリアスのハッシュテーブルエントリーの検索によって決定されるように）まだエイリアスでフェッチされていないような次のエントリーを選定する。
フェッチ済みのエントリーに記録し、これらのエントリーの以後の使用を防止する。
ミューテックスを解放する
206
選定されたウェッブページをフェッチする
208
ミューテックスを要求して待つ
210
212
フェッチは成功したか？
Y
N
218
ミューテックスを解放する。
214
（入力ファイルの）ディスクファイルのエントリーに受け取ったリターンコードに基づくフェッチ失敗エラーインジケータを記録する
216
（入力バッファの）ディスクファイル及び対応するハッシュテーブルエントリーのエントリーにフェッチフラグをセットする。フェッチされたウェッブページを解析する（詳細は図４を参照）

FIG. 4A

【図４】

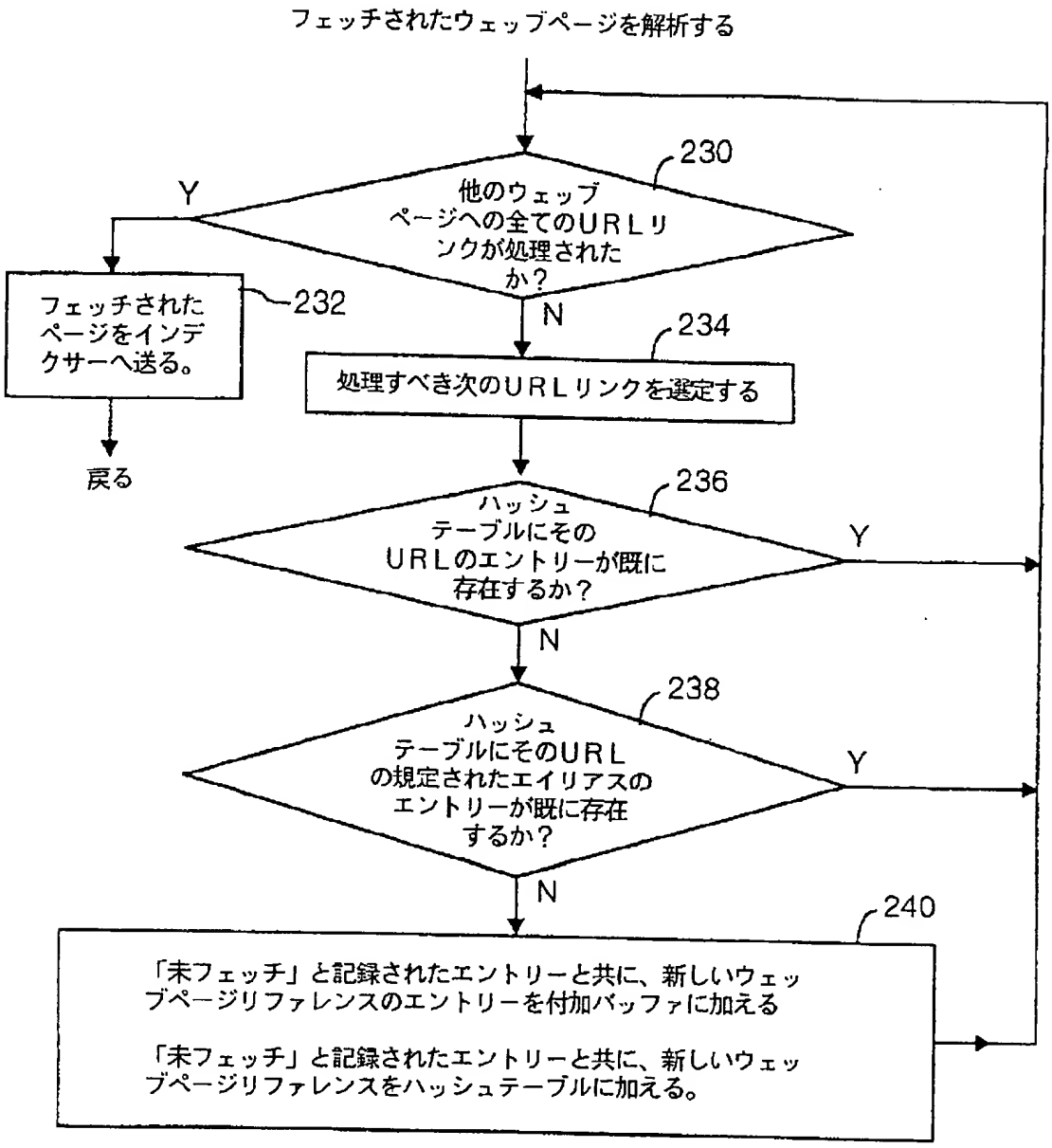

FIG. 4B

FIG. 4A
FIG. 4B

FIG. 4

【国際調査報告】

INTERNATIONAL SEARCH REPORT

Inter mal Application No
PCT/US 96/19831

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F17/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	PERSONAL COMPUTER MAGAZINE, JAN. 1996, VNU BUSINESS PUBLICATIONS, UK, pages 90-92, 94, 97 - 98, 100, XP000646762 SIMPSON D ET AL: "The searchers World Wide Web search engines" see the whole document ---	1,6,11, 16
A	ELECTRONIC LIBRARY, OCT. 1995, LEARNED INFORMATION, UK, vol. 13, no. 5, ISSN 0264-0473, pages 467-476, XP000647883 SHA V T: "Cataloguing Internet resources: the library approach" see the whole document -----	1,6,11, 16

☐ Further documents are listed in the continuation of box C.

☐ Patent family members are listed in annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
4 April 1997	16.04.97

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Katerbau, R

Form PCT/ISA/210 (second sheet) (July 1992)

フロントページの続き

(81)指定国　　EP(AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AP(KE, LS, MW, SD, SZ, UG), AM, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, DE, DK, ES, FI, GB, GE, HU, JP, KE, KG, KP, KR, KZ, LK, LT, LU, LV, MD, MG, MN, MW, NO, NZ, PL, PT, RO, RU, SD, SE, SI, SK, TJ, TT, UA, UZ, VN

【要約の続き】

ーケンシャルに行われ、単一のＩ／Ｏオペレーションとして、シーケンシャルディスクファイルからの多数のエントリーが入力バッファへ移されるようにする。従って、シーケンシャルディスクファイルは入力バッファからアクセスされる。同様に、シーケンシャルファイルに加えられるべき全ての新しいエントリーは付加バッファに記憶され、付加バッファが一杯になった時はいつでも、付加バッファの内容はシーケンシャルファイルの最後に加えられる。このようにして、ウェッブインフォメーションディスクファイルへのランダムアクセスは排除され、ディスクアクセス制限によって引き起こされる待ち時間は減少される。